Directory Structure for the Archival of Large Amount of Data Files

Dr. Martin P. Lévesque

Defence Research Establishment, Valcartier

Technical Note

DREV TN-2002-002

2002-12-17

Author

Dr. Martin P. Lévesque

Approved by

Dr. Jean-Marc Garneau

Section head / Space Optronics

© Her Majesty the Queen as represented by the Minister of National Defence, 2002

© Sa majesté la reine, représentée par le ministre de la Défense nationale, 2002

Abstract

This document present a simple method to archive data. It consists principally in a method of naming directories within a pre-defined directory structure and where these directory names correspond to keywords that can be retrieved from a database. No special software neither file format are required, only a few discipline from the operators that make the data acquisition. However, the benefits are considerable. First, it becomes easier to exchange data with others and, second, the archive can be automatically decoded by a program. This directory structure allows to program a database application that can read the archive content and automatically feed a database catalogue with all the information retrieved about the data files found in the archive. This last point is particularly useful for large deployment trial where thousand of data files are acquired.

Résumé

Ce document présente une méthode simple pour archiver des données. Elle consiste principalement en une méthode de nomenclature de répertoires utilisée à l'intérieur d'une structure de répertoires prédéfinis et ces noms de répertoires correspondent à des mots clefs qui peuvent être retrouvés dans une banque de données. Aucun logiciel spécial ni aucun format de fichier ne sont requis, seulement un peu de discipline de la part de l'opérateur qui fait l'acquisition des données. Cependant, les avantages sont considérables. Premièrement, il est plus facile d'échanger des données avec les autres et, deuxièmement, l'archive peut être décodée automatiquement par un programme. Cette structure de répertoires permet de programmer une application de base de données qui peut lire le contenu d'une archive et automatiquement remplir le catalogue de la base de données avec toutes les information retrouvées sur les fichiers de données trouvés sur ladite archive. Ce dernier point est particulièrement pratique pour les essais à grand déploiement où des milliers de fichiers de données sont acquis.

This page intentionally left blank.

Executive summary

During trials and military missions, it becomes more and more common to acquire thousands of data files acquired by several different sensors which are operated by independent teams from different countries. The data becomes easily a complete mess and nobody can retrieve and analyze the global information. Usually, the analyst inspects only the data acquired by its own team and ignores the data acquired by others because of the difficulty of matching the different pieces of information.

The ultimate solution would be to apply database techniques to encode, filter and organized the information. However, this is often an overkilled solution and nobody used it because of the complexity of the database, the lack of training and more particularly the excessive workload that represent the data capture. Once the issue of data capture is solved (which requires a lot of resources), it should be easy to use the database (by untrained users) with pre-programmed forms. The overkilled sensation related to the database approach principally origins from the fact that all the data files (and dependent information) have to be encoded and inserted (by the analyst) into the database while these information are used (retrieved) only once in a while. If the data handling is completely done manually, the workload (of the data capture) can completely overcome the benefits that the database could provide.

Therefore, the solution resides in an intermediate solution which simply consists in a method of data archiving where the data files are well ordered. The idea is simply 1) to pre-define a directory structure that should be used by every trial attendees and 2) naming the directories with significant names, names that have the value of keywords since they can eventually be associated with database entry points.

This method requires only a minimal discipline from the users (no special technology or software constraints) but the advantages are considerable. It makes easier the information exchanges (more comprehensive archives) and the retrieval of information from archives generated by others. However, the most important advantage is that this archiving method prepares the data for an eventual automatic loading into a database. Because of the well ordered directory structure, an auto-loader program could decode the directory names and do all the job of database insertion, which represents the biggest effort in the data handling.

This technical note explains and illustrates how the data acquired by sensors should be ordered and saved in archives. It describes the directory structures, the classes of keywords and their relation with a database. It also gives examples of complementary information that can be retrieved about a specific data files when the directory keyword names are well defined.

Lévesque, M. P., 2002, "Directory Structure for the Archival of Large Amount of Data Files", DREV TN 2002-002, Defence Research.

P518527.PDF [Page: 8 of 37]

Sommaire

Durant les essais et missions militaires, il devient de plus en plus courant d'acquérir des milliers de fichiers de données acquis par différents capteurs opérés par des équipes indépendantes provenant de différents pays. Les données deviennent facilement un fouillis complet et personne ne peut retrouver ni analyser l'information globale. Habituellement, l'analyste inspecte seulement les données acquises par sa propre équipe et ne tient pas compte des données des autres à cause de la difficulté d'apparier les différents éléments d'information.

La solution ultime serait d'appliquer les techniques de base de données pour encoder, filtrer et organiser l'information. Cependant, c'est souvent une solution excessive et personne ne l'utilise à cause de la complexité des bases de données, le manque d'entraînement et plus particulièrement la charge de travail excessive que représente la saisie de données. Lorsque le problème de la saisie de données est résolu (lequel exige beaucoup de ressources), ce devrait être facile d'utiliser la base de données (par un usager non entraîné) avec des formulaires préprogrammés. La sensation de solution excessive associée à l'approche des bases de données provient principalement du fait que tous les fichiers de données (et les informations qui en dépendent) doivent être encodés et insérés (par l'analyste) dans la base de données, alors que ces information ne sont utilisées que de temps en temps. Si l'entretient des données est complètement faite à la main, la surcharge de travail (de la saisie de données) peut complètement outrepasser les bénéfices que la base de données pourrait produire.

Donc, la solution réside dans une solution alternative, laquelle consiste simplement en une méthode d'archivage de données où les fichiers de données sont bien ordonnés. L'idée est simplement 1) de prédéfinir une structure de répertoires qui devrait être utilisée par tous les participants aux essais et 2) désigner les répertoires par des noms significatifs, des noms qui ont la valeur de mots clefs puisqu'ils peuvent être associés éventuellement avec des points d'entrés d'une base de données.

Cette méthode requiert seulement un minimum de discipline de la part des usagers (aucune technologie spéciale ni contrainte de logiciel), par contre les avantages sont considérables. Elle rend plus facile les échanges de données (archives plus compréhensives) et la recherche d'information dans les archives générées par les autres. Cependant, l'avantage le plus important est que cette méthode d'archivage prépare les données pour un éventuel téléchargement automatique dans une base de données. À cause de la structure de répertoire bien ordonnée, un programme de chargement automatique peut décoder les noms des répertoires et faire tout le travail d'insertion dans la base de données, lequel représente le plus gros effort de maintenance des données.

Cette note technique explique et illustre comment les données acquises par les capteurs devraient être ordonnées et sauvegardées sur les archives. Il décrit les structures de repertoires, les classes de mots clefs et leurs relations avec les bases de données. Il donne aussi des exemples d'information complémentaire qui peuvent être retrouvée sur les fichiers de données particuliers quand les noms clefs des répertoires sont bien définis.

Lévesque, M. P., 2002, "Directory Structure for the Archival of Large Amount of Data Files", DREV TN 2002-002, Centre de recherches pour la défense, Valcartier.

P518527.PDF [Page: 9 of 37]

Table of contents

Abstract i

Executive summary iii

Sommaire iiv

Table of contents v

List of figures vii

1. Introduction 1

2. Data Acquisition Context and Data Archival 3
   - 2.1 Acquisition Context and Directory Structure 3
     - 2.1.1 An Example of a Structured Archive 4
   - 2.2 Possible Directory Structure and Their Identification Codes 5

3. Archive Database 6
   - 3.1 Inserting Information Into the Database 6
   - 3.2 Information Encoded Into the Database by the Analyst Before the Automatic Decoding of the Archive 6
   - 3.3 Archive Decoding and Subscription of Files Into the Database Catalogue 7
     - 3.3.1 The Database Content 7
     - 3.3.2 Decoding of Information Containe4d in the Directory Names 8
   - 3.4 Decoding of Information Contained in the File Header 9
   - 3.5 Extraction of Information From the Data Analysis 10

4. Special Directories and Files 11
   - 4.1 Measurement Documentation 11
     - 4.1.1 Identifying the Comment Files by the Auto-Loader 11
   - 4.2 Sensor Configuration. 'Sensor.config.txt' File 12
     - 4.2.1 Content of the 'Sensor.config.txt' File 12
       - 4.2.1.1 Keywords Identifying a Component type 13
       - 4.2.1.2 Mnemonic Identifying a Sensor Component 13
       - 4.2.1.3 Special Action Executed by the Auto-Loader Program 13
     - 4.2.2 Zone of Application of the 'Sensor.config.txt' File and Local Overwrite 14
   - 4.3 The Description Directories 15
     - 4.3.1 The Mnemonic Description Files 16
   - 4.4 The Calibration Measurements 17
     - 4.4.1 Laboratory Calibrations 17
     - 4.4.2 Live Calibrations Acquired by the Sensor 17

P518527.PDF [Page: 10 of 37]

4.4.2.1 Range of Application of the Calibration Measurement 18
4.4.2.2 Calibration Overwrite 18
4.4.2.3 Linking Data Calibration into the MUSIMS Database 19

5. Conclusion 20

6. References 22

Distribution list 23

List of figures

Figure 1. Data archiving and retneval system 1

Figure 2. Context of data acquisition and suggested archive structure 4

Figure 3. Overview of the MUSIMS database tables 8

Figure 4. Keywords (from the directory structure) decoded (by the auto-loader program) and linked to the MUSIMS database tables 9

Figure 5. Text file that documents a measurement file 11

Figure 6. Action executed by the auto-loader on the measurement and documentation files 12

Figure 7. Occurrence of the configuration files and local overwrite possibilities for an 'OSTDEC' archive structure 14

Figure 8. Special names for archiving information other than sensor measurements 15

Figure 9. Example of insertion of calibration data through other measurements 18

Figure 10. Example of local calibration overwrite 18

Figure 11. Representation of relation between data file in the MUSIMS database 19

List of tables

Table 1. Keyword classes 5

Table 2. Best Directory Structure 5

Table 3. Identification keywords and component mnemonics 13

Table 4. Type of information that could be stored in 'description' directories 16

1. Introduction

For decades now, people have acquired data which is underused because this data is not share and finally lost for many raisons. Very often, a sensor measurement is analyzed on the fly during its acquisition and never inspected again. This data is stored on an archive that accumulate dust until nobody remember the content of this archive or until the archive medium is so old that no device can read it again. If only other people had known that this data exist, someone else could have used it for a different purpose or for the same original goal but with different analysis tools.

The goal of this work is to present a method to structure data archives. This method allows a maximum of data documentation, dissemination and reused with the minimal effort from the person who make the data acquisition. This paper presents only a directory structure sufficiently organized (and in the same time adaptable) to allow to retrieve data by everybody and even by programs. This structure allows to encode information (keywords) directly into the directory names and identify the context (sensor, owner, circumstances, etc) of acquisition. It can then be decode later (by a database engine) and external knowledge can be automatically linked with the data. This process can automatically documents the data files and add value to the data. Otherwise, the data would stay anonymous piece of valueless data.

This archival structure is only a part of a more global approach (Fig. 1) that contains also a database, a an autoloader (a program that can decode the directory names and makes automatic database insertions and connections) and browser. This system, called MUSIMS (for Multi Sensor Information Management System), will be described in future publications.

One may note that the methodology presented here is only an archival methodology, not a database one. A database engine can be used later to create catalogue and for browsing, but the first step is to make order in the data collection. Effectively, in this report, the archive structure will always be presented in a context of a database approach (which gives a sense to the proposed structure), but even in the absence of database engine, the archival method can be very useful to help people to recover information in the context of data exchange.

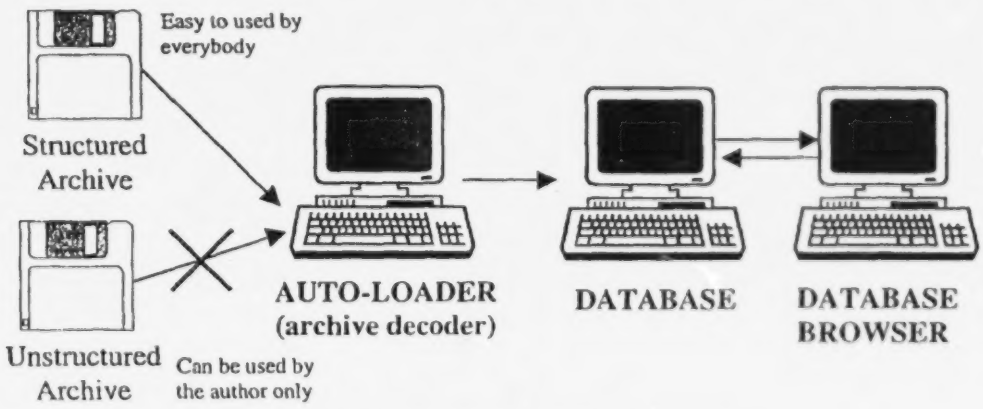


Figure 1 – Data archiving and retrieval system

To summary, the clear advantages of the use of this method to structure archives are:

- that it is easier to:
  - exchange data,
  - understand data from other,
  - select, filter or retrieve data,
- and it allows to encode complementary information
  - by using significant keywords to name directories (identification of sensor, trial, etc.),
  - by using special file or directory names (calibration, sensor parameters, etc.),
- and it makes uniform the structure and prepare data for automatic post-processing by machines. (dedicated program).

This work has been done in the context of data acquisition and handling for large deployment trials where sensors and experiments are numerous, various and heterogeneous. Representatives from DND attend very often to these kinds of trials and a huge amount of data is generated and not well handled. The method presented here is certainly not a solution to all issues, but it certainly better than the void that actually exists in data management.

This work has been achieved between April 2000 and June 2001 under project 5ea13, 'Space Base EO/IR System Analysis'.

2. Data Acquisition Context and Data Archival

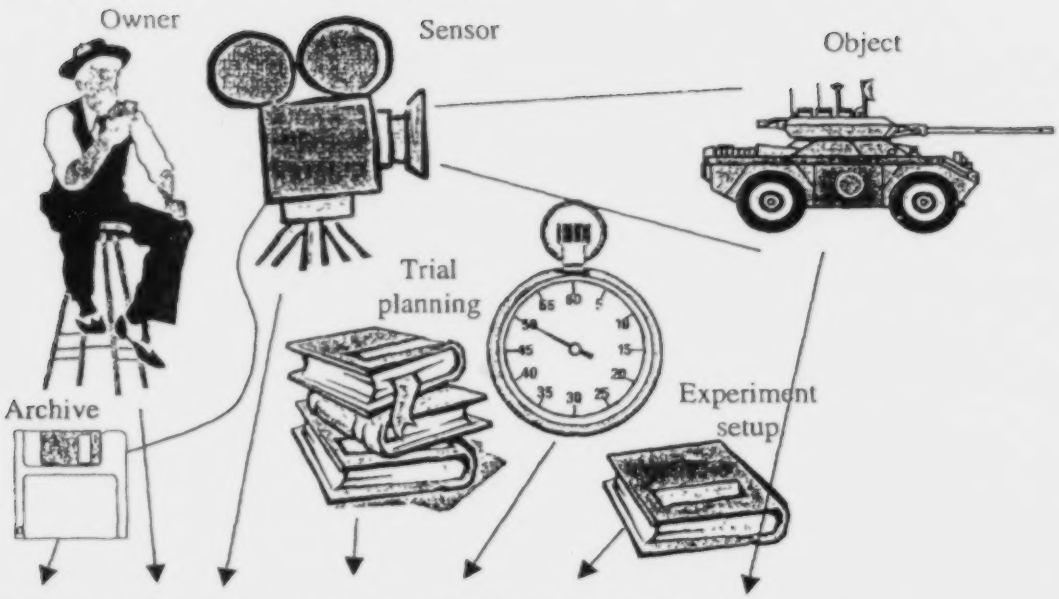
Usually, people gives names to data files which have more or less meaning. Very often, the names have a sense only for the sensor operator, and this, most of the time, for a limited period of time. This is a cause of major issue for large scale trials where a lot of people operate many different sensors, measuring many objects, simultaneously or not. When time comes to exchange data (with calibration and documentation), this is a complete mess. Nobody use the same terminology and acronym and it is very painful to link different piece of data together. Moreover, even for the data owner it is difficult to remember, after a certain time, the context of acquisition (scene setup, ground truthing, etc.) of a specific measurement.

Therefore, there is a need for an archival methodology used by everybody, but this restriction will have to be kept as simple as possible, otherwise people will not use it. Hence, it has been decided not to impose a data format or a file name convention. The only thing that can be imposed is the use of a certain number of identification keywords which can be used to link the data with other knowledge that can be retrieved in an external database. For example, this knowledge can be the sensor description, the acquisition circumstances, trial planning and ground truthing, etc. Moreover, these keywords are used as directory name in such a way that all data acquired by a sensor, or during a specific experiment, or something else, can be easy to isolate and retrieve. Hence, people continue to acquire data as they were used to but the files are stored in an imposed directory (or folder).

This imposed directory structure has two beneficial effects. First, people can retrieve data more easily, event if acquired and stored by foreign operators. Second, it is possible to create a decoding program (and this has been done and tested but not documented and published yet) that can automatically insert the data in a database and also create automatically links with knowledge already stored. Hence, by interrogating the database, it becomes easy to get any kind of information related to a specific data file such as the sensor characteristics, the trial manager's address, the description of the observed object and so on. Furthermore, this allows to sort the data accordingly to the needs by selecting the appropriate parameters. These parameters can be as sophisticated as wished because by programming search algorithms in the database (which operate using all possible related knowledge) everything is possible. Just consider this possible search: "*search only IR images of this object acquired simultaneously from this and this points of view*", to see how the indirect information can be managed.

2.1 Acquisition Context and Directory Structure

Fig. 2 illustrates the information components involved in the process of data acquisition in the context of an organized trial. These components suggest the list of most common keyword classes (listed in TABLE I) that can be used and the most obvious directory structures.



VOLUME:/ OWNER / SENSOR / TRIAL / DATE / EXPERIMENT / SCENE / measurements /. .

Figure 2 – Context of data acquisition and suggested archive structure

2.1.1 An Example of a Structured Archive

The directory structure presented in Fig.2 has already been used with success with the data acquired by DREV at the MUST2000 trial (Ref.1). More than 8000 data files were stored on a DVD and the auto-loader program was able to entirely decode the archive, record the information in the MUSIMS database and links the measurement with the contextual information.

Here is an example of such a file:

MUST-2K-001:/CA_DREV/VISIR_AGEMA/MUST2000/2000MAY25/ . .
. SERIAL011/LAV-1/agw-aslav-aus-lighy.bmp

This example illustrates a case where only the sensor operator knows the meaning of the file name. In this case, "agw-aslav-aus-lighy.bmp" means: **AG**ema infrared camera with the **W**ide field of view optic measuring the **AU**Stralian **L**ight **A**rmor **V**ehicle with a **AUS**tralian **LIGH**t (color) camouflage net. Now, if you do not know the operator, then there is nothing to do with such a name.

Now, if the keyword structure is decoded, here is the information one can get:

MUST-2K-001 ·	name of the DVD archive,
CA_DREV	owner organization·
	CAnada, Defense Research Establishment Valcartier
	(allows to retrieve full contact address),
VISIR_AGEMA.	name of the sensor
	(allows to retrieve sensor attributes),
MUST2000	trial name

	(allows to retrieve information from the trial planning),
2000MAY25	date,
SERIAL011	name of the experiment (allows to retrieve object status and ground-truthing),
LAV-1	object acronym which identify which vehicle is being observed (can also be a site or event acronym). Combine with the experiment description, one can find the complete object status.

From these keywords, once can retrieve where (GPS information) was the vehicle LAV-1 (there was also other vehicles) and what was its status (engine on, Australian camouflage net on, etc.) from the description of the experiment "SERIAL011" of the trial "MUST2000". Furthermore, the characteristic of the sensor "VISIR_AGEMA" are entirely known and a special file "sensor_config.txt" (described in next chapter) provides the information that the wide FOV optic was in used. All these information can be retrieved by everybody from the database.

2.2 Possible Directory Structures and Their Identification Codes

The order of directories is not really important as long that a single structure is used for an entire archive, but different archives can have different structures accordingly to the user needs. This obliges only to specify to the decoder what is the directory structure used by the archive. TABLE II illustrates what is an "OSTDEC" (used in the previous example) or a "TOSDEC" archive structure code. As explained in TABLE II, different people may require different structures. The only constraint is that the analyst has to start the auto-loader program with the appropriate code to get the appropriate decoding sequence.

TABLE I
Keyword classes

Keyword class code	Meaning	Comments
O	Owner	name of the sensor-owner organization
S	Sensor	sensor name or nickname
T	Trial	name of the trial
D	Date	encoded date (yymmdd)
E	Experiment	name of the schedule experiment
C	sCene	object, site or event name

TABLE II
Best Directory Structures

Directory structure	Data sorted by:	Comments
OSTDEC	Owner/Sensor/. .	good for the sensor operator
TOSDEC	Trial/...	good for the trial planners
CDOS	object/. .	good for the analyst

3. Archives and Database

Since the ultimate goal to create structured archives is to be able to automatically upload information that describe their contents into a database, let us see what look like such a database in order to have a better understanding of the system.

Before to inspect the database approach, let say that it is not necessary to have such databases to apply the method of data archival. By itself, the archive structure allows to make order in the data and make easy to recover information. For example, with a UNIX system, one can filter files by using a command like 'ls /*/*/MUST2000/*/*/LAV*/*.jpg'. For a 'OSTDEC' archive, this last UNIX command line means: list all JPEG images of every LAV vehicles acquired by every sensors during the MUST2000 trial. Similar (but less selective) searches can be done using the command 'find' provided with other operation systems (on Macintosh and PC). However, for more complex queries, which can also involves other information not contained in the directory names, database engines are required.

3.1 Inserting Information Into the Database

For every good database design, there is at least three distinct steps for the information decoding and inserting (into the database). They are:

1 - collateral information gather and inserted by the analyst,
2 - archive decoding (keywords found in directory structure) and creation of the *catalogue* into the database with a first level of linking with information already entered by the analyst and
3 - file decoding (by reading the file header content) and insertion/update of information into the database.

For a more sophisticated systems which have also the capability of data processing (using a depository of processing algorithms), one can add:

4 - extraction of information by processing the data and insertion of this newly found information into the database.

3.2 Information Encoded Into the Database by the Analyst Before the Automatic Decoding of the Archive

Hence, before to be able to automatically decode any archive, the database needs to be already filled with information that the auto-loader program will recognized later. Therefore, the analyst has to complete first the database tables where the directory-keywords and other mnemonics will be found. For instance, he will have to insert (into the database) the information about:

- the sensors (along with their component description, calibration, configuration, etc.),
- the sensor owner (point of contact, address, etc),
- the trial (point of contact, location, date, etc.),
- the experiment (scene setup, time, location, objects, object status, etc.) and
- every other relevant information.

Thereafter, when the auto-loader program will decode keywords in the directory structure, it will be able to retrieve them from the database and links the data file with other knowledge.

The next chapter indicates how these information can be recorded in the archive using special files and directories in order to be 1) easy to be read by the analyst, 2) not processed as measurement files by the auto-loader.

3.3 Archive Decoding and Subscription of Files Into the Database Catalogue

The next step consist in starting the auto-loader program and have it read the archive and register all the measurement files into the database catalogue. Doing this, the auto-loader program also makes the links with the knowledge recorded by the analyst. In order to have a better understanding of what happened during this step, lets us have an overview of the database design and decoding process.

3.3.1 The Database Content

Figure 3 illustrates the basic content of the MUSIMS database. The complete database design is much more complex, but this figure allows us to see which kind of information is stored.

The primary requirement imposed to the MUSIMS database is that it must keep track of huge amount of off-line information. There is not limit to the total archive capacity since all the data do not have to be resident on the active computer storage area. Like a library, MUSIMS can manage huge room filled with off-line archive without considering any limit of storage capability. Therefore, the first task performed by the auto-loader program is simply to recognize the archived files that contain measurement data and register these files to the main catalogue. Therefore, this table (CATALOGUE) is the master piece of the database while the branch 'DATA_FILES' contains the information about where to find these files (including copies and backups).

The database has also several branches where information can be stores by themes as illustrated in Fig. 3. This figure gives a good overview of the database content. It is expected that a complete description will be published later in a separate report.

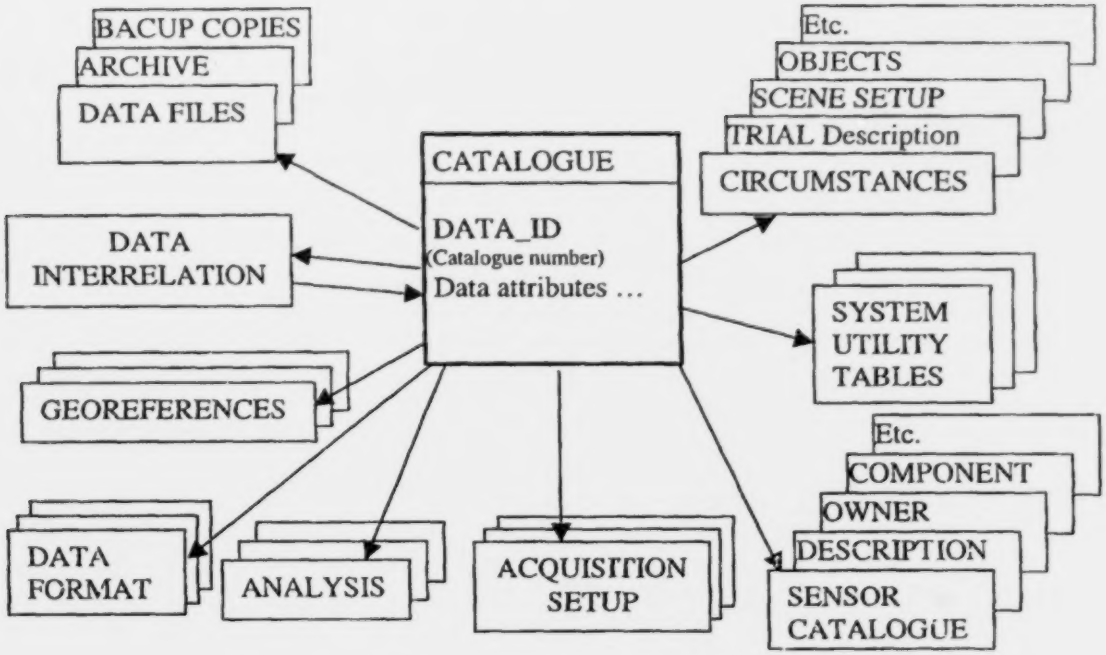


Figure 3 – Overview of the MUSIMS database tables

3.3.2 Decoding of Information Contained in the Directory Names

Once the database has been filled with the basic information describing the sensors, trials, experiments and others, the auto-loader can begin to automatically decode the archives and records the data files into the catalogue.

The decoding of the structured archives involves three process which are:

1- decoding, identifying and referencing the directory keywords,
2- detecting the measurement data file and registering them into the catalogue and
3- updating the catalogue by linking the data (file) to the entities previously identified by the directory keywords.

In the first step, a directory names are decoded in relation with their hierarchic level and the archive structure code (e.g. OSTDEC). Hence, if the third directory level is a 'S' keyword type (i.e. a Sensor name), the auto-loader will retrieved the foreign key from the table SENSOR that points to this sensor. During this step, a validation and disambiguation process is applied. Some disambiguation process may required to consider group of keyword simultaneously. The auto-loader is cleaver enough to retrieve the right sensor or object even if many different objects have the same name (e.g., many sensors may have the same name but with different owner or many objects may have the same mnemonic but defined in different trials, etc.) This decoding process is illustrated in Fig. 4 where keywords (or groups of keywords) are related to database tables.

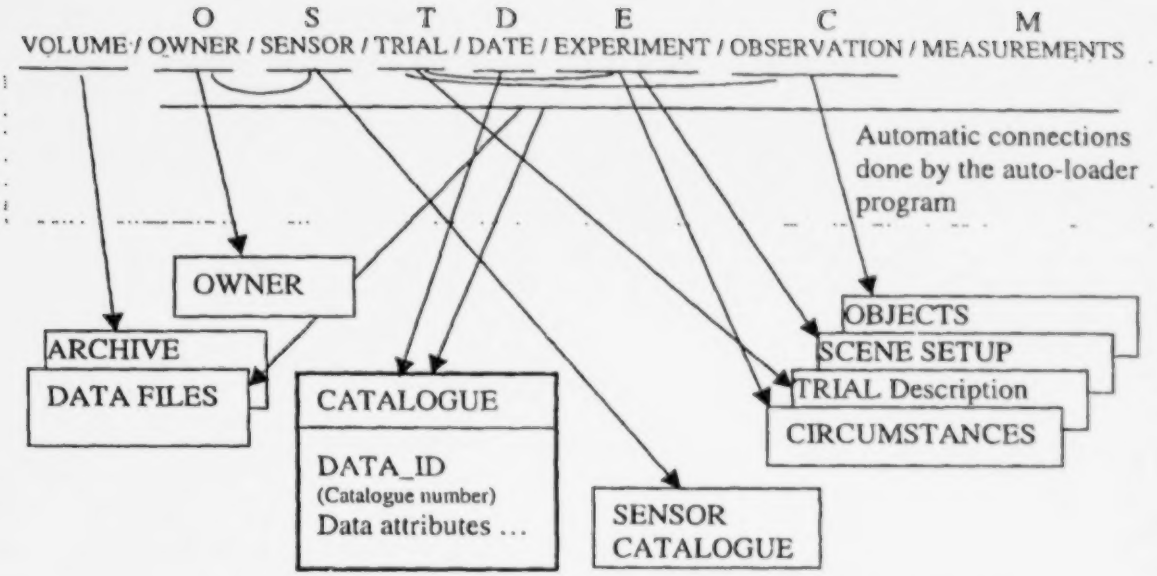


Figure 4 – Keywords (from the directory structure) decoded (by the auto-loader program) and linked to the MUSIMS database tables

Once the directory structure is completely decoded (one branch at the times, eventually they will all be explored and decoded), the auto-loader looks for measurement entities (most of the time a single file, but some sensors generate many files for a single measurement). It is cleaver enough to find and recognize the measurement files because, somewhere in the database, it is known which kind of file is generated by a specific sensor (for example a specific digital camera generates ".jpg" files, i.e. a JPEG images while another generates '.bmp' i.e. a bitmap images). This allows to ignore the other files which are not measurement data but most of the time comments.

When a measurement entity is found and identified, it is registered in the catalogue by according it a new catalogue number (DATA_ID). The complete knowledge of the data location is recorded in the table branch called "DATA_FILES". These tables store the information necessary to retrieve the data file (or group of files for a multiple-file measurement), including multiple copies or backup the database application software may have done.

Finally, the table 'CATALOGUE' is update with all information (i.e. foreign keys) that has be found (by decoding the directory keywords) that concern the measurement entity 'DATA_ID'. This also includes relation with other measurement entities like calibration measurements (described in next Chapter)

3.4 Decoding of Information Contained in the File Header

The next step would consist in opening the data files and extract information form their headers. However, this requires to know the file format and have the appropriate software library to decode it.

It is not the intention of the MUSIMS auto-loader program to do such a decoding. In fact, it only decodes the directory structure, register the data files and make a few links into the database catalogue. This allows to register files with unknown format (unknown by the program but probably known by the analyst) in a first step, leaving the job of header decoding to another program in a subsequent step. Hence, this approach allows to gather a maximum of information, even if all the information is not immediately decodable

However, such a decoding program has already been partially developed with the ACIDE (Auto Context Image Database Exploitation) project (Ref. 2). This decoder had access to a repository of header decoders. After having identified the header type and properly decoded it, it had the capability to reorganize the extracted parameters into thematic groups of variables or packets Then it could communicate these packets to standard methods that upload this information in the database. Hence, the first part of the information extraction is header-type dependent while the last part is database design dependent. Thus, it was easy to update the program to 1) introduce new header-type decoders and 2) to update the upload methods accordingly to the database design modifications.

3.5 Extraction Information From the Data Analysis

Ultimately, information could be extracted by data processing and the obtained knowledge could be used to update the database. The protocol to automatically adapt and apply processing algorithms to the incoming data files has already been described in the ACIDE project (Ref. 3). This protocol took into account the diversity of sensors, data types, scenarios, needs, etc. The idea was to use an algorithm repository along with a processing manager that contains scripts (or expert system rules, also called 'agent' in the 'blackboard' context) able to recognize the acquisition context of the data files and link these files to pre-established scenario of processing.

4. Special Directories and Files

During the archive structure decoding process, it is foreseen that some specific names will receive special attention. Since an archive does not contain only measurement data files, it is important to give the appropriate processing to these names of exception. It is foreseen that an archive can also store information such as:

1 - comments about a measurement data file,
2 - documentation of the sensor setup,
3 - general description and documentation of:
- sensors,
- organizations,
- trials
- experiments
- objects,
- etc. and

4 - measurement calibrations.

The following sections shows how these information can be encoded with simple conventions and recognized and process by an auto-loader program.

4.1 Measurement Documentation

The most common file that accompanies measurement are the documentation or comment text file. The convention used to identify them, as shown in Fig. 5, is that the root name is the same as a measurement file but the file extension is change for a known extension of a text file such as a '.txt' or a '.doc' file. This allows to add extra information saved along with the data files on the archives. For example, Fig. 5 illustrates the case where an image file (0411227.jpg) acquired by a CCD camera is documented with a text file (0411227.txt)

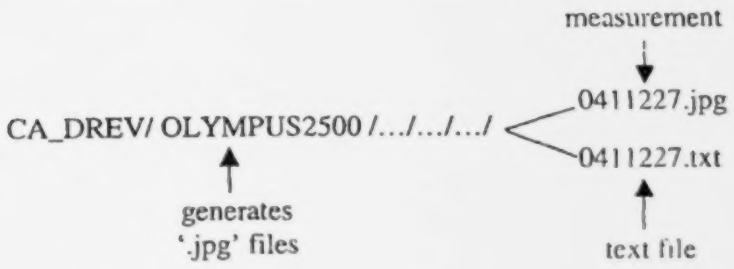


Figure – 5 Text file that documents a measurement file

4.1.1 Identifying the Comment Files by the Auto-Loader

When decoding a measurement directory, the auto-loader program is able to make the difference between measurement files and other files. It already knows the type of measurement files to record into the database catalogue. It got this information from the description of the sensor (from the database) which itself was obtained by the decoding of the 'SENSOR' keyword. For instance, Fig. 5 indicates that the auto-loader can retrieved from the database that the sensor

OLYMPU2500 (the one which belong to the CA_DREV organization) is a CCD camera that generates JPEG images with the file extension '.jpg'.

Once the auto-loader has found a '.jpg' file in a measurement directory, it looks for text file having the same root name. The measurement '.jpg' file first generates a new entry in the database catalogue (illustrated in Fig. 6) and after the text file is read and inserted into the 'COMMENTS' field of the 'CATALOGUE' table. Hence, not only the measurement is documented in the archive, this documentation is also available in the database.

One may note that only the content of the text file is read and inserted into the database. The measurement file is not loaded, only its existence (i.e. the file and archive names and their locations) are recorded in the catalogue. Usually, the measurement are Mbytes files while comments are only a few bytes. This allows a computer to manage a total volume of information thousand times bigger than its storage capacity.

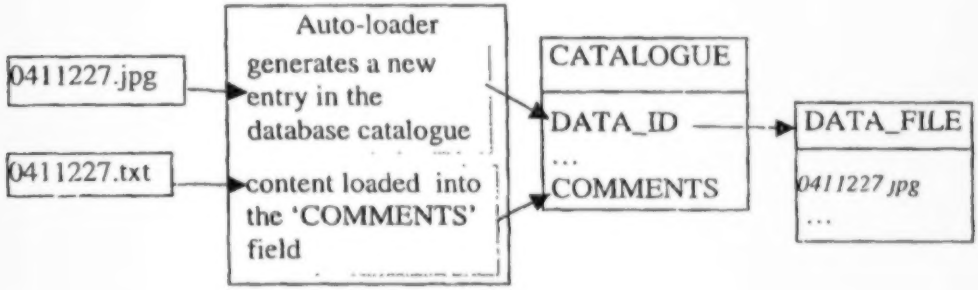


Figure 6 – Action executed by the auto-loader on the measurement and documentation files

4.2 Sensor Configuration: 'Sensor.config.txt' File

A very useful file, which can be inserted almost everywhere in the archive structure, is the one that documents the sensor status. Since it is important for post-acquisition processing to know the status of the sensor, information can be recorded here by the operator to inform the analyst (or the auto-loader program) about the sensor settings.

Usually, sophisticate sensor auto-documents the data by inserting parameters into the file header For example, it is common that geo-referenced sensors encode GPS position, time and pointing parameters along with the measurement data. However, there are some operation that the acquisition system cannot automatically record and the information must be encoded by the operator. This is the case, for example, when the operator change the camera optic, filter or another component and that the system does not have the appropriate interface to record this event. Therefore, the operator can create a file called 'sensor.config.txt' (by convention) where these set-up are recorded. The auto-loader program will be able to recognized and process these files. Other file names can be used (rather than 'sensor.config.txt'), but the analyst has to update the information in the utility tables of the database (see Fig. 3).

4.2.1 Content of the 'Sensor.config.txt' File

Basically, the 'sensor.config.txt' file is an ASCII file that contains two columns of keywords. First, a universal keyword defines the class of component that is being modified and,

second, a mnemonic that is specific to the concerned sensor. Table III shows an example 'sensor.config.txt' file content.

TABLE III
Identification keywords and component mnemonics

Identifying keyword	Sensor component mnemonic example 1	Sensor component mnemonic example 2
OPTIC	2x, 4x	WFOV, NFOV, 75mm
DETECTOR	dct1	InSb, CCD
DETECTOR-B	B	InSb
INPUT_PUPIL	pupil-a	7mm
OUTPUT_PUPIL	pupil-b	8mm
FILTER	red	3um

4.2.1.1 Keywords Identifying a Component Type

In every cases, it is mandatory to use one of the pre-defined keywords to identify the type of sensor component that is being to be changed. These keywords are listed in the left column of Table III, but, depending on the sensor specification, other component types could be added in the future. Also, only the components that are being changed need to be specified (not necessary to list all the sensor components). Using pre-defined keywords allows analyst (and the auto-loader program) to recognize the situation and react accordingly. For example, a 'sensor.config.txt' file may contains only the sentence 'OPTIC = 2x' and the auto-loader program will be able to retrieve the so called optical component (for this sensor) targeted by this change.

4.2.1.2 Mnemonic Identifying a Sensor Component

However, the mnemonics that modify the status of the considered component are specific to the sensors (they are defined by the operator). In the MUSIMS database, each sensor can point to a number of components listed in the table "COMPONENT_CATALOGUE" (not described in this report), which itself points to more detailed tables. Each component is identify by a primary key and also by a mnemonic. The mnemonic alone may not be unique, but it must be unique when associated with the sensor identification (for example, there may be many optics call "2x", but the VISIR_AGEMA has only one called "2x"). These mnemonics are defined by the sensor operator only once and are inserted into the database for future references. Also, it is a good idea to document and save the mnemonic definitions along with the sensor description. Hence, other analysts or operators will have a chance to catch their meaning. This point is discussed in Section 4.3.1.

4.2.1.3 Special Action Executed by the Auto-Loader Program

Once a 'sensor.config.txt' file is found, the auto-loader program reads the known (left side) keywords and, accordingly to these keywords, finds the references to the sensor components indicated by the (right side) mnemonics. Now, the auto-loader is able to make the right links between the measurement data file and the sensor with the appropriate setup. For example, in a future usage of the database, not only the user will be able to know than the '341456.jpg' image was acquired by the 'OLYMPUS2500', he will also be able to know that the '2x' optic was used.

It has already been mentioned (in Fig. 3) that a link is created between the measurement data and the sensor that made the acquisition. This information was provided by the directory keyword 'SENSOR'. In fact, the process is more sophisticate and accurate because not only the sensor is identified but also its configuration. In the table SENSOR, a sensor is recorded with all its possible configurations. It is identified by a combination of its primary key 'SENSOR_ID' and the variable 'CONFIGURATION_NUMBER'. When a measurement is recorded into the database and that the sensor is known but not its status, the data file is linked with the default sensor description (CONFIGURATION_NUMBER=0) where the components are unidentified (unknown). But if the components are identified (using a 'sensor.config.txt' file), then the measurement can be linked with a more accurate sensor description where parameters, calibration and other characteristics are known in relation with this sensor configuration. This is usually very important for post processing such as data calibration, interpretation, analysis and processing.

For a better understanding of the decoding process, here is exactly what the auto-loader program does. First, the auto-loader program finds the foreign key 'OWNER_ID' by retrieving the directory-keyword 'OWNER' in the table 'OWNER', Then, it retrieves the foreign key 'SENSOR_ID' from the table 'SENSOR' by finding in the list of sensors (which belong to OWNER_ID) the one that has the nickname defined by the directory-keyword 'SENSOR'. By default, the CONFIGURATION_NUMBER is set to zero, the value for which all sensors are defined without considering their components. These sensor descriptions are the less accurate. When a 'sensor.config.txt' file is encountered, the auto-loader retrieves the foreign keys (depending on the keyword defined in the file) from the tables OPTIC, DETECTOR, FILTER and PUPIL (using the mnemonic defined in the file and the value of SENSOR_ID which uses these components) and updates the variable CONFIGURATION_NUMBER for the one which corresponds to this sensor setup. If only certain components are changed, the others maintain their last known values. Finally, the measurement file is recorded into the database table CATALOGUE and a link is made toward the table SENSOR with the foreign key SENSOR_ID and the appropriate value for the variable CONFIGURATION_NUMBER.

4.2.2 Zone of Application of the 'Sensor.config.txt' File and Local Overwrite

Many configuration files can be present in the archive for a same sensor. Obviously, they do not have the same range. Basically, the configuration file that can be applied to a measurement is the first one that can be found when exploring up the directories. An example is presented in Fig. 7.

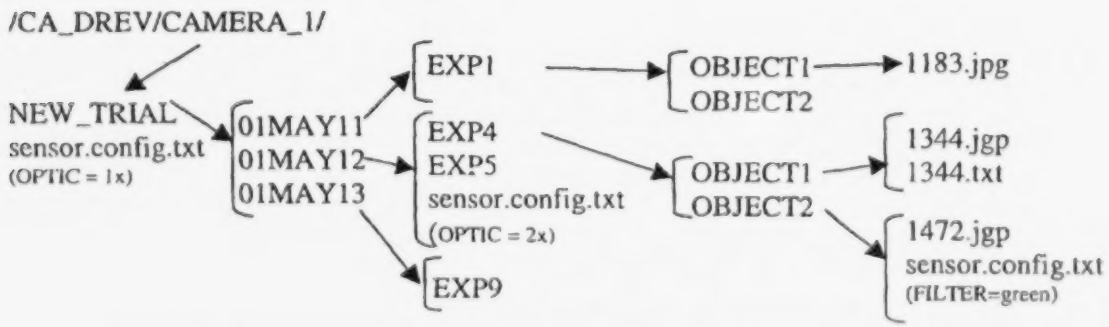


Figure 7 – Occurrence of configuration files and local overwrite possibilities for an 'OSTDEC' archive structure

In this example, one can see that the default setup for the entire trial (NEW_TRIAL) for the sensor called CAMERA_1 is that the optic '1x' is used This setup is applicable to the image 1183.jpg. However, during day of May 12th 2001, the operator change the optic for the '2x' one, but the '1x' was still used on May 13th. Thus, image 1344.jpg (which is also documented) is acquired with the '2x, optic. Finally, when acquiring the object 'OBJECT2' during the experiment 'EXP4' on May 12th, a green filter was also used for the image 1472.jpg.

This example illustrates that the encoding of information in the archive structure is easy. Operators may leave notes that analysts can retrieve and that the auto-loader can automatically decode and insert in the database. However, one may note that this method is only an alternative for the sensors system that do not encode these information into the file headers.

4.3 The Description Directories

The archive can also contains general information that describe the sensors, trials, experiments, objects, etc. These information are very useful for the analyst but are not measurement data files that must be automatically loaded by the auto-loader program into the database. Therefore, certain names are reserved for these files or directories that the auto-loader program will simply ignore. The are the 'documentation(s)' and 'description(s)' files or directories.

Figure 8 illustrates the use of such directories when mixed with the other keywords of the archive structure. For example, one can see that the sensor called 'VISIR_AGEMA' participated to two trials. There is also a directory 'description' beside the two trial. It is obvious that the description of each trial should be saved in their respective directory so this 'description directory' applies to the sensor. For example, it may contains photos of the sensor, text files that describe it, result of calibrations done in laboratory, etc. Similarly, all the other 'description' directories contain information that are applicable to the keyword defined by the father directory.

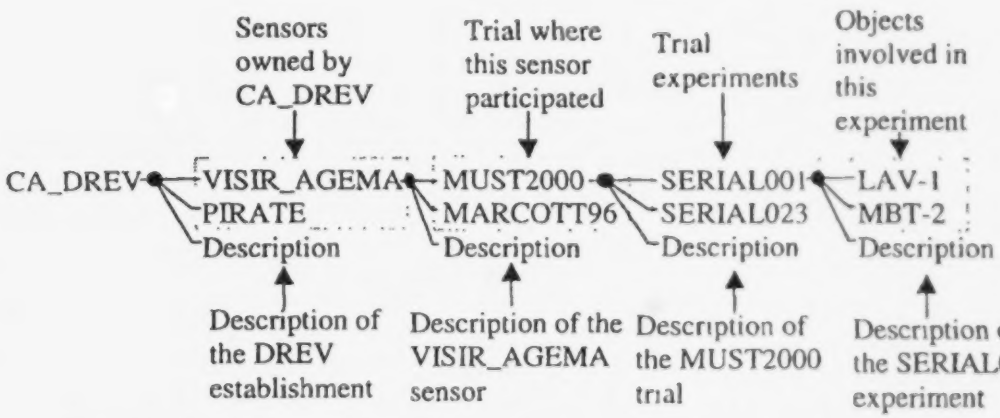


Figure 8 – Special names for archiving information other than sensor measurements

Table IV indicates the kind of information that should be stored in the 'description' directories. As mentioned above, these directory will not be decoded by the auto-loader program.

However, most of the time, the database analyst will have to consult them to update the database with the lists of mnemonics and keywords that will be encountered later in the directory structure. Moreover, he will have to do this before to submit the archive to the auto-loader program. If not, there is a chance that the auto-loader decoder will not be able to recognize the experiment, object and sensor component mnemonics.

TABLE IV
Type of information that could be stored in 'description' directories

Description of...	Suggested content of the description directory
OWNER	Point of contact, address, email , etc.
SENSOR	Photos illustrating the sensor, laboratory calibration, list of characteristics, sensor components, list of component identification mnemonics, etc.
SENSOR/ COMPOMENT	Photos illustrating the component, laboratory calibration, list of characteristics, etc
TRIAL	Point of contact, list of experiments, objects (vehicles, calibration patterns, etc.), sites (with GPS coordinates and ground description). event (official diner, someone birthday, etc,) with mnemonic definition used in the trial.
EXPERIEMENT	Description of the experimental setup, object locations and status, etc.
SCENE	Ad-hoc information on what is contained in the field of view

4.3.1 The Mnemonic Description Files.

To identify sensor components, objects, experiment names, a number of mnemonics (and keywords) need to be known before the load the archive into a database. This is the case, for example, with the component mnemonics found in the 'sensor.config.txt' files (section 3.2) or the object name found in directory keywords 'SCENE'. It would be a very good idea to list and defined these mnemonics in files and save these files under the 'description' directories. Hence, before loading the archive, the analyst could inspect manually the archive to find mnemonics that are not defined yet in the database and make the necessary updates. This is particularly important when he receive data acquired by a sensor he did not know before

To illustrate the need for mnemonic descriptions, here are the files that could be used:
- under the 'trial/description' directory, the trial planners could create files that define and document the mnemonics used for the experiment, event, object and site used during the trial,
- under the 'sensor/description' directory, the sensor operator could defined the mnemonic used to identify components,

- under the 'owner/description' directory, the list of sensor keywords could be provided, -
- etc.

4.4 The Calibration Measurements

In the archive structure, a special attention has been given to the calibration measurements. There are two kinds of calibration. The first kind consists in laboratory measurements, usually not acquired by the sensor itself but rather by other pieces of equipment. The second kind consists in controlled measurements acquired by the sensor itself.

4.4.1 Laboratory Calibrations

The laboratory calibrations are measurements that are not acquired very often (sometimes only once in the sensor lifetime) but they are mandatory to evaluate the sensor performances and adapt post-processing methods to the data acquired by this sensor. Since these information are used to characterized the sensor, it is recommended to save them under the directory 'SENSOR/DESCRIPTION' Among the calibrations done in laboratory, The most common one that one may thing about are:

- LSF: Line spread function (optic characterization),
- PSF: Point spread function (optic characterization),
- optical distortions (optic characterization),
- MRTD: Minimum Resolvable Temperature Difference (IR system),
- NETD: Noise equivalent Temperature Difference (IR system),
- linearity response (detector),
- homogeneity (detector),
- dead detectors (detector),
- spectral response (detector, optic, system),
- electronic noise,
- detection threshold/saturation

and others, accordingly to the nature of the sensor.

4.4.2 Live Calibrations Acquired by the Sensor

A live calibration (or a in-the-field calibration) usually consists in measuring a known object (a calibrated blackbody, a test pattern, etc.) in order to be able to deduce the sensor characteristics, the atmospheric transmission and other phenomena. These measurements are then used as reference to calibrate the following measurements.

Therefore, it is important to give a special place to these calibrations in the directory structure to be able to link them with the other dependent measurements. This is why it is foreseen that the special name 'Calibration' (or 'calib' or 'calibrations') is reserved in the archive directory structure.

Hence, the calibration measurement can be saved in the 'Calibration' directory. Then, everybody (and program) will know the importance of this measurement and know how they can be linked with the other measurements found in the same directory level or below this directory such as shown in Fig. 7. In this example, all images of target 1 and 2 acquired during the experiment 6 and 7 can be calibrated using the data of the calibration measurements done on may 26th.

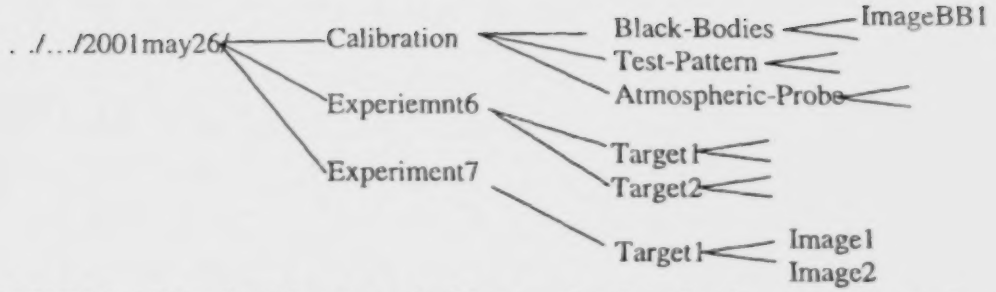


FIGURE 9 – Example of insertion of calibration data through other measurements (here, the calibration is valid for every experiments that occur that day)

4.4.2.1 Range of Application of the Calibration Measurement

Depending on the valid range of the calibration, the measurements can be saved at different levels. The frequency for the acquisition of calibration measurements depends on the sensors and the nature of the experiments. Some sensors are calibrated only one in their life time while other sensors require a calibration before each measurement.

Hence, depending on the calibration needs, a calibration measurement can be:

- done in laboratory (for permanent characteristics) and saved under the directory SENSOR/DESCRIPTION
- done before the trial and saved under the directory SENSOR/TRIAL
- done before each experiment and saved under the directory TRIAL/EXPERIMENT
- done daily and saved under the directory DATE
- done before each measurement and saved under the directory EXPERIMENT/SCENE or under other subdirectory if necessary.

4.4.2.2 Calibration Overwrite

However, it is possible to have a calibration valid for a long period of time and that suddenly a new calibration is required for a specific measurement. In such a case a local overwrite is allowed. In fact, the way the calibrations and measurements are linked together is the closest calibration found in the same or higher directory levels is the one that is applied to the measurement. Hence, in Fig. 10, the 'taget1' in 'experiment7' has its own calibration while all the other measurements use the calibration saved in the directory '2001may26/Calibration'.

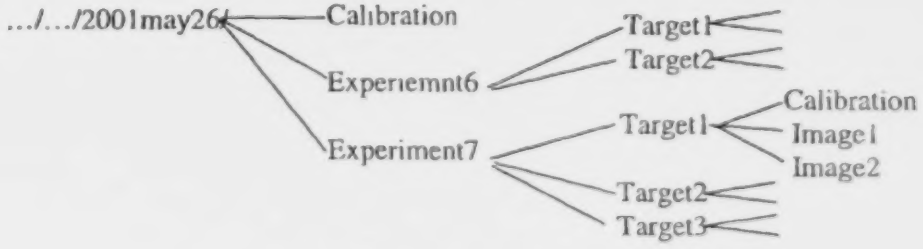


Figure 10 – Example of local calibration overwrite

4.4.2.3 Linking Data and Calibration into the MUSIMS Database

From the data point of view, there is no difference between a calibration measurement and a normal measurement. Hence, in the MUSIMS database, both are recorded as measurement done by the sensor.

However, when a 'calibration' directory name is encounter, a special flag is raised and after the calibration measurement files are normally recorded into the database catalogue. After that, for all other normal measurement data files recorded into the catalogue, a special relation is declared (see Fig. 11) that links these files to the appropriate calibration files. This relation is defined by the relation 'relation=calibration' in the table DATA_INTERRELATION. MUSIMS allows also other kind of relations (such as description, illustration, etc.) but they are not the object of this report. Thereafter, if one wants to calibrate data included in a files referenced by the catalogue, this other table (DATA_INTERRELATION) will provide the information to retrieve the appropriate calibration files.

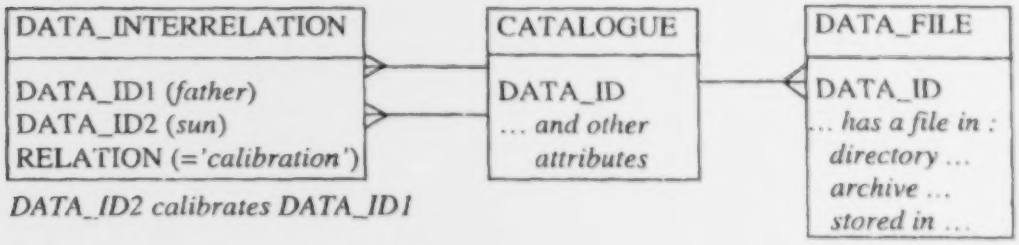


Figure 11 – Representation of relations between data files in the MUSIMS database

One may note that a calibration set may consists in many files and that many measurement files can point to this calibration set. This is called a 'n-to-n' relation in the database and it is totally allowed.

5.0 Conclusion

It is difficult to impose restrictions or a new way of doing something to people that have their habits well established. Therefore, the archiving method presented here (i.e. the method of directory naming presented in Chapter 2) imposed only the minimal possible adaptation. People continue to acquired data the same way they are used to. They are not required to change file format or anything else. They only have to record their data under a simple directory structure. Furthermore, it is very obvious to identify the owner, sensor and trial in the directory names. Only the experiment and scene (sites, objects and events) names impose a certain discipline. In fact, they should be provided by the trial organizers before the trial in order to have everybody use the same names.

The benefits of this method (easier to exchange, document, retrieve data, etc.) are well presented in the introduction. For the small effort it requires (during the acquisition), it is worthwhile to well structure the data archive, particularly when one consider the possibility of automatic loading (in a database), referencing and, eventually, processing of the data. The linking of information (sensor, trial and object descriptions) to the data increases a lot its value.

Furthermore, the presented method is very adaptable. It allows the user defines the structure he prefers. The only condition is that it is not allowed to have different structure used on the same archive (constrains provided by the auto-loader program).

The major benefits of using the archiving method is the possibility to use an auto-loader program which can upload the knowledge of complete archive directory structure into the database catalogue. Usually, this step of data handling requires days or even weeks of works from the analyst, but the prototype developed for the MUSIMS project was able to register in the catalogue the 8000 measurement data files (acquired by 11 different sensors) contained in a DVD/R in only a few minutes. Thereafter, the user had the possibility to browse the data using a sophisticate database search engine (ORACLE) along with a sophisticate browser tool developed with ORACLE / FORM. This browser had also the capability to load data files from a juke box (that had the capability to load off-line archive into the reader) and view the data using the appropriate viewer (this information is also contained in the database) and all of this using a local network because the juke box and the database were on different machines than the user one.

The last important point of the archive structure proposed here is the possibility to encode complementary information (like sensor setup, measurement comments, description, etc.) that can be read by others and by the auto-loader. Also, and probably the most important one, is the way that calibration measurements are clearly identified and automatically linked to the other measurements (in Chapter 4). However, these points are secondary and may not be used if the user does not desire them (but that would be regrettable) as long as the directory naming method presented in Chapter 2 is still applied.

Therefore, view the simplicity of the method, view the benefits it brings in term of value added to the data and the possibility for the automatic archive decoding and uploading into database, this archiving method is recommended for all large deployment trials where a lot of heterogeneous sensors and targets and a lot of people with different culture are involved.

Also, the use of database tools are not mandatory to give a sense to this archiving methods. Simply the fact of sharing a common archiving method make easier to exchange data between the trial participants. Therefore, why not to use it? And while using an archiving method, why not using one that is also compatible with database technology?

6. References

1. MUST2000 Trial, TTCP SEN TP-4 / AG-5 (final report not published yet).

2. Lévesque Martin P., "*Auto Context Image Database Exploitation (ACIDE): Description of the ACIDE_IO.PGK Module* (U), DREV TM 1999-136 (Protected B), 64 pages

3. Lévesque Martin P., "*Auto Context Image Database Exploitation (ACIDE): System Concepts and Definitions* (U), DREV TM 1999-135 (Protected B), 36 pages.

4. List of Distribution

INTERNAL DISTRIBUTION

TN 2002-002

1 – Director General
3 – Document Library
1 – M.P. Lévesque (Author)
1 – T. Smithson
1 – J.-M. Garneau
1 – J.-P. Ardouin
1 – D. St-Germain
1 – L. Sévigny
1 – J. Maheux
1 – J. Boiteau
1 – P. Lahaie
1 – S. Buteau
1 – J. Lévesque
1 – A. Jouan
1 – J.-M. Thériault
1 – J. Dumas

EXTERNAL DISTRIBUTION

1 – DRDKIM (unbound copy)

SANS CLASSIFICATION

COTE DE SÉCURITÉ DE LA FORMULE
(plus haut niveau du titre, du résumé ou des mots-clefs)

FICHE DE CONTRÔLE DU DOCUMENT

1 PROVENANCE (le nom et l'adresse)	2 COTE DE SÉCURITÉ (y compris les notices d'avertissement, s'il y a lieu)
Centre de la recherche pour la défence, Valcartier 2459 Boul. Pie XI nord, Val-Bélair, QC G3i 1X5	sans classificaton

3. TITRE (Indiquer la cote de sécurité au moyen de l'abréviation (S, C, R ou U) mise entre parenthèses, immédiatement après le titre.)
Directory Structure for the Archival of Large Amount of Data Files

4 AUTEURS (Nom de famille, prénom et initiales. Indiquer les grades militaires, ex. Bleau, Maj Louis E)
Lévesque, Martin P.

5 DATE DE PUBLICATION DU DOCUMENT (mois et année)	6a NOMBRE DE PAGES	6b. NOMBRE DE REFERENCES
Janvier 2002	22	3

7. DESCRIPTION DU DOCUMENT (La catégorie du document, par exemple rapport, note technique ou mémorandum. Indiquer les dates lorsque le rapport couvre une période définie.)
Memorandun

8. PARRAIN (le nom et l'adresse)

9a. NUMÉRO DU PROJET OU DE LA SUBVENTION (Spécifier si c'est un projet ou une subvention)	9b. NUMÉRO DE CONTRAT
5ea13	

10a. NUMÉRO DU DOCUMENT DE L'ORGANISME EXPÉDITEUR	10b. AUTRES NUMÉROS DU DOCUMENT
	N/A

11. ACCÈS AU DOCUMENT (Toutes les restrictions concernant une diffusion plus ample du document, autres que celles inhérentes à la cote de sécurité.)

- [x] Diffusion illimitée
- [ ] Diffusion limitée aux entrepreneurs des pays suivants (spécifier)
- [ ] Diffusion limitée aux entrepreneurs canadiens (avec une justification)
- [ ] Diffusion limitée aux organismes gouvernementaux (avec une justification)
- [ ] Diffusion limitée aux ministères de la Défense
- [ ] Autres

12. ANNONCE DU DOCUMENT (Toutes les restrictions à l'annonce bibliographique de ce document. Cela correspond, en principe, aux données d'accès au document (11). Lorsqu'une diffusion supplémentaire (à d'autres organismes que ceux précisés à la case 11) est possible, on pourra élargir le cercle de diffusion de l'annonce)

SANS CLASSIFICATION

COTE DE LA SÉCURITÉ DE LA FORMULE
(plus haut niveau du titre, du résumé ou des mots-clefs)

SANS CLASSIFICATION

COTE DE LA SÉCURITÉ DE LA FORMULE
(plus haut niveau du titre, du résumé ou des mots-clefs)

13 SOMMAIRE (Un résumé clair et concis du document. Les renseignements peuvent aussi figurer ailleurs dans le document. Il est souhaitable que le sommaire des documents classifiés soit non classifié. Il faut inscrire au commencement de chaque paragraphe du sommaire la cote de sécurité applicable aux renseignements qui s'y trouvent, à moins que le document lui-même soit non classifié. Se servir des lettres suivantes: (S), (C), (R) ou (U). Il n'est pas nécessaire de fournir ici des sommaires dans les deux langues officielles à moins que le document soit bilingue.)

This paper present a simple method to archive data. It consists principally in a method of naming directories within a pre-defined directory structure and where these directory names correspond to keywords that can be retrieved from a database. No special software nether file format are required, only a few discipline from the operators that make the data acquisition. However, the benefits are considerable. First, it becomes easier to exchange data with others and, second, the archive can be automatically decoded by a program. This directory structure allows to program a database application that can read the archive content and automatically feed a database catalogue with all the information retrieved about the data files found in the archive. This last point is particularly useful for large deployment trial where thousand of data files are acquired.

Ce papier présente une méthode simple pour archiver des données. Elle consiste principalement en une méthode de nomenclature de répertoires utilisé à l'intérieur d'une structure de répertoires pré-définit et ces noms de répertoires correspondent à des mots clés qui peuvent être retrouvés dans une banque de données. Aucun logiciel spécial ni aucun format de fichier ne sont requis, seulement un peu de discipline de la part de l'opérateur qui fait l'acquisition des données. Cependant, les bénéfices sont considérables. Premièrement, il est plus facile d'échanger des données avec les autres et, deuxièmement, l'archive peut être décodée automatiquement par un programme. Cette structure de répertoires permet de programmer une application de base de données qui peut lire le contenu d'une archive et automatiquement remplir le catalogue de la base de données avec toutes les information retrouvés à propos des fichiers de données trouvés sur ladite archive. Ce dernier point est particulièrement pratique pour les essaues à grand déploiement où des milliers de fichiers de données sont acquis.

14. MOTS-CLÉS, DESCRIPTEURS OU RENSEIGNEMENTS SPÉCIAUX (Expressions ou mots significatifs du point de vue technique, qui caractérisent un document et peuvent aider à le cataloguer. Il faut choisir des termes qui n'exigent pas de cote de sécurité. Des renseignements tels que le modèle de l'équipement, la marque de fabrique, le nom de code du projet militaire, la situation géographique, peuvent servir de mots-clés. Si possible, on doit choisir des mots-clés d'un thésaurus, par exemple le "Thesaurus of Engineering and Scientific Terms (TESTS)". Nommer ce thésaurus. Si l'on ne peut pas trouver de termes non classifiés, il faut indiquer la classification de chaque terme comme on le fait avec le titre.)

Archive, database, structure, directory, keywords, sensor, trial, mass storage

#518527

CA 1792

SANS CLASSIFICATION

COTE DE SÉCURITÉ DE LA FORMULE
(plus haut niveau du titre, du résumé ou des mots-clefs)